

Data PREPROCESSING

22F-3190

22F-3093

**How Embeddings are Performed**

After completing the cleaning phase of the dataset, the next important task was to create meaningful embeddings. These embeddings were needed so that machine learning models could easily understand and process the data later on.

After completing the cleaning phase of the dataset, the next important task was to create meaningful embeddings. These embeddings were needed so that machine learning models could easily understand and process the data later on.

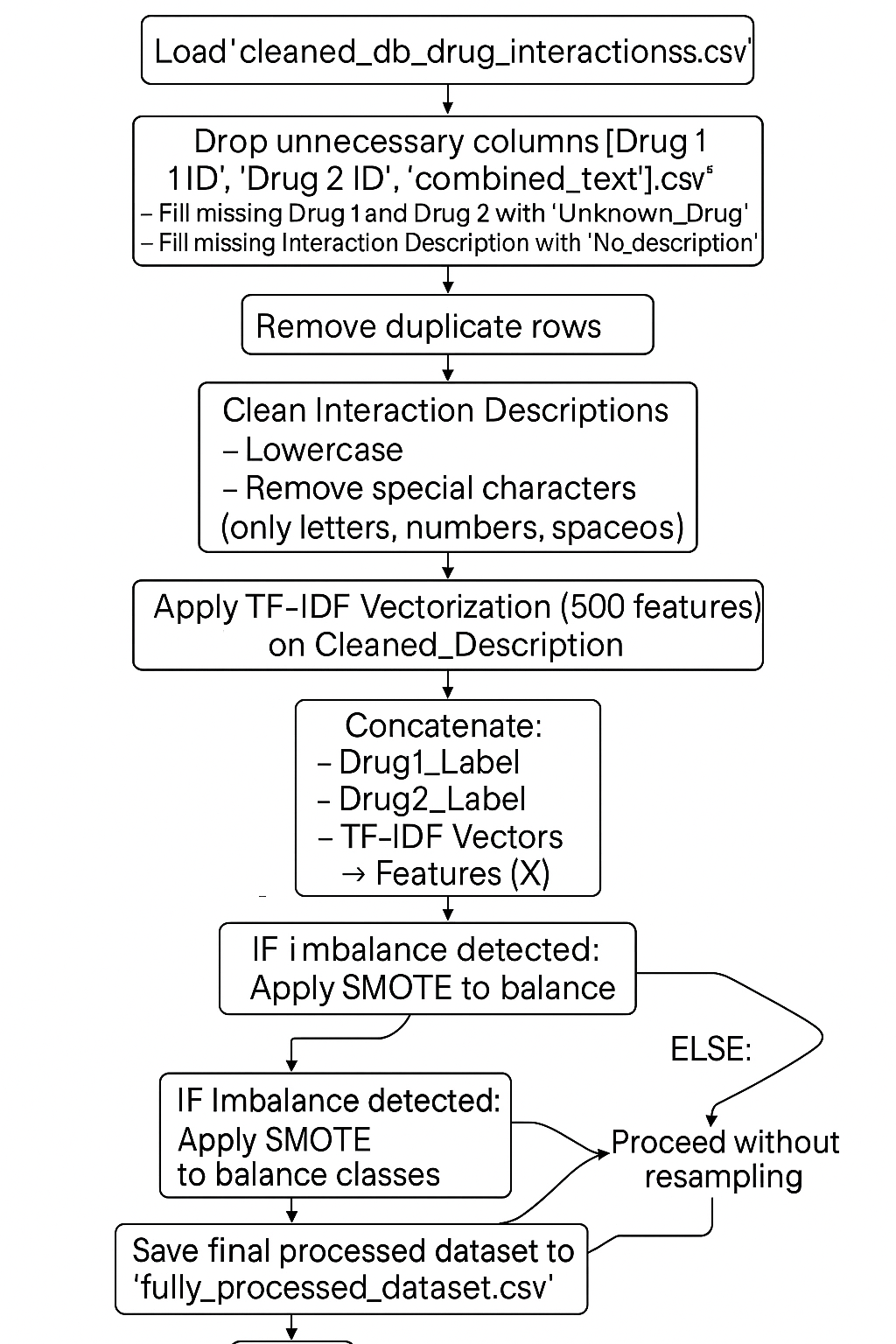
The first thing I did was to apply Label Encoding to the two drug columns and the interaction descriptions. Label Encoding is a simple but effective way to turn text into numbers without making the feature space unnecessarily large. Since the number of unique drugs and interaction types was quite high, using something like one-hot encoding would have exploded the number of features, making the dataset bulky and hard to handle.

Next, I focused on the interaction descriptions, which are basically free-text fields. I used a TF-IDF Vectorizer to convert these descriptions into numerical form. TF-IDF (Term Frequency-Inverse Document Frequency) helps in giving more importance to rare but meaningful words while ignoring common words that don’t add much value. I limited the TF-IDF to the top 500 features and filtered out common English stopwords to make sure only the important keywords were captured.

After that, I combined both the Label Encoded features and the TF-IDF vectors into one big feature set. I used horizontal stacking for this, meaning that I just lined up the label numbers and TF-IDF scores side by side for every sample. This gave a rich representation that included both the structured information (drug IDs) and the unstructured information (descriptions).

Another thing I had to check was whether some interaction types had way fewer samples compared to others. If there was a major imbalance, I used a method called SMOTE (Synthetic Minority Oversampling Technique) to artificially create new examples for the minority classes. This way, the model would not become biased towards the more common types of interactions.

At the end of this whole process, I had a final feature matrix ready — neatly balanced, rich in information, and perfectly suited for feeding into machine learning models.



**Motivation**

The main reason for choosing this specific embedding process was the nature of the data itself. Since the dataset contained a mix of structured information (drug names) and unstructured text (interaction descriptions), I needed a way to handle both effectively. Label Encoding worked well for the categorical drug names because it kept things simple and compact. For the descriptions, TF-IDF made sense because it captured the essence of the text without overcomplicating the model with heavy NLP techniques. Combining the two allowed me to represent each sample completely. Also, by applying SMOTE where needed, I made sure the model would not be biased toward the majority classes. Altogether, this approach was practical, efficient, and well-suited to the goal of building a strong predictive model.

**Preprocessing**

The very first thing I did was load the dataset and quickly check for basic issues — things like missing values, unnecessary columns, and any weird anomalies. It’s always important to get a sense of what you’re dealing with before you start cleaning.

I noticed that there were some columns like 'Drug 1 ID', 'Drug 2 ID', and 'combined\_text' that didn’t really add much value to the prediction task. So I simply dropped them to keep the dataset clean and focused.

Next, I tackled missing values. Instead of deleting rows, which would mean losing potentially useful data, I filled missing drug names with a placeholder "Unknown\_Drug" and missing descriptions with "No\_description". This way, no data points were wasted, and the model would still get a chance to learn something useful from those records.

Another important thing was to remove duplicate rows. Duplicates can mess up model training because they bias the model toward repeated patterns. So I made sure every row was unique after this step.

Then came text processing. Since drug names sometimes had typos, I used TextBlob to fix spelling errors. I know TextBlob isn’t perfect, but it’s lightweight and good enough for catching small mistakes automatically. After that, I cleaned the text in the interaction descriptions by converting everything to lowercase and removing special characters, leaving only alphanumeric content. This standardization makes it much easier for models to learn meaningful patterns later.

For the encoding part, I used Label Encoding again — one for each drug and one for the interaction type. This turned all the categorical information into a numeric format.

The descriptions were transformed separately using a TF-IDF Vectorizer, but this time focused only on the cleaned version of the text. Limiting the TF-IDF to 500 important words kept the feature space small but still meaningful.

Finally, I checked if the dataset was imbalanced in terms of interaction types. If it was, I used SMOTE to generate synthetic examples for underrepresented classes. Otherwise, I moved forward without applying any balancing.

At the end of it all, I saved the cleaned and processed dataset into a new CSV file called ‘fully\_processed\_dataset.csv’. With this final version, I knew the data was clean, balanced, well-structured, and totally ready for model training.

